REMARKS

The examiner is thanked for the performance of a thorough search.

By this amendment, Claims 1, 27, and 53 are amended, and no claims are added or canceled. Hence, Claims 1-3, 5-29, 31-55, and 57-78 are pending in the application.

The amendments to the claims as indicated herein do not add any new matter to this application. Furthermore, amendments made to the claims as indicated herein have been made to exclusively improve readability and clarity of the claims and not for the purpose of overcoming alleged prior art.

Each issue raised in the Office Action mailed April 15, 2008 is addressed hereinafter.

## I. ISSUES RELATING TO THE CITED ART

Claims 1-3, 5-10, 12, 14-16, 18-29, 31-36, 38, 40-42, 44-55, 57-62, 64, 66-68, and 70-78 stand rejected under 35 U.S.C. § 103(a) as allegedly being unpatentable over U.S. Patent No. 6,675,382 issued to Foster ("*Foster*") in view of Kon et al., "Dependence Management in Component-Based Distributed System," January 2000, IEEE ("*Kon*"). This rejection is respectfully traversed.

Claims 11, 13, 17, 37, 39, 43, 63, 65, and 69 stand rejected under 35 U.S.C. § 103(a) as allegedly being unpatentable over *Foster* in view of *Kon*, and in further view of U.S. Patent Publication No. 2004/0003266 to Moshir et al. ("*Moshir*"). This rejection is respectfully traversed.

A. CLAIM 1

Claim 1 recites:

> A method of dynamic installation and activation of software packages in a node in a distributed network of nodes, the method comprising the computer-implemented steps of:

> storing, in a software package storage of a master node in the distributed network, a plurality of software packages and a plurality of software modules that the nodes in the distributed network will be using;
> wherein each software package of the plurality of software packages contains at least one module and associated dependency information;
> receiving a software update for a node on said master node;
> wherein the software update contains a set of one or more software packages;
> storing the software update on said software package storage;
> wherein said master node notifies said node that a software update is being requested;
> wherein said master node passes said node identities of one or more software packages to be updated and module dependencies;
> **wherein said node determines, using the module dependencies and before installing the software update, running processes on said node that will be affected by the software update**. (emphasis added)

At least the above-bolded elements of Claim 1 are not taught or suggested by *Foster* or *Kon*, either individually or in combination. The Office Action concedes that *Foster* "does not explicitly disclose that said node determines, using the module dependencies, running processes on said node that will be affected by the software update and said master node notifies said node that a software update is being requested" (page 4). The Office Action then cites FIGs. 2 and 4 and the first paragraph under section "Dynamic Dependencies" on page 4 of *Kon* for disclosing "wherein said node determines, using the module dependencies, running processes on said node that will be affected by the software update" as recited in Claim 1 (pages 4-5). This is incorrect. The cited first paragraph merely states:

> In our model, a component configurator manages each component. The component configurator is responsible for storing the runtime dependencies between a specific component and other system and application components. Depending on the implementation, a component configurator might be able to refer to components running on a single address space, on different address spaces and processes, or even on different machines in a distributed system. (emphasis added)

Thus, a component configurator stores runtime dependencies and is able to refer to components running (a) on a local machine, (b) on multiple address spaces of a machine, or (c) on a remote machine. Based on this cited portion, the Office Action appears to equate: (1) the component

configurator of *Kon* with the recited node of Claim 1 and (2) a component of *Kon* with the recited software update. Even if these correlations could be made, the component configurator of a particular component, at best, identifies components that the particular component depends on and components that depend on the particular component. The component configurator identifies these dependencies at runtime, i.e., after the component has been installed. In contrast, the recited node in Claim 1 determines what processes **will be** affected by a software update **before** the software update is installed. Fundamentally, *Foster* and *Kon* fail to teach or suggest, both individually and in combination, that a node determines running processes on the node that will be affected by a software update **before** the software update is installed.

Based on the foregoing, because *Foster* and *Kon* fail to teach or suggest, individually and in combination, all the features of Claim 1, Claim 1 is patentable over *Foster* and *Kon*. Reconsideration and withdrawal of the rejection of Claim 1 under 35 U.S.C. § 103(a) is therefore respectfully requested.

B. CLAIMS 27 AND 53

Independent Claim 27 is a computer-readable storage medium claim and independent Claim 53 is an apparatus claim. Each of Claims 27 and 53 recite features discussed above that distinguish Claim 1 from *Foster* and *Kon*. Therefore, each of Claims 27 and 53 is allowable for the reasons given above with respect to Claim 1.

C. DEPENDENT CLAIMS

The dependent claims not discussed thus far are dependent claims, each of which depends (directly or indirectly) on one of the independent claims discussed above. Each of the dependent claims is therefore allowable for the reasons given above for the claim on which it depends. In

addition, some of the dependent claims introduce additional limitations that independently render them patentable.

*1. Claim 3*

For example, Claim 3 depends on Claim 1 and additionally recites:

> wherein each node has a list of desired characteristics stored on said master node which is compared by said master node to each module in the software update to determine which modules in the set of one or more software packages should be sent to a node.

The Office Action cites col. 7, lines 35-38; col. 8, lines 34-36; and col. 11, lines 28-48 of *Foster* for disclosing Claim 3. This is incorrect. These cited portions merely state:

> Control file 220 includes "control information" that describes the contents of payload file 210, operations that may be performed on them, and other information regarding the history of creation or installation of those files.
>
> The OSVERSION and PLATFORM fields specify the operating system version and the system platforms over which the embodiments of invention can be executed.
>
> In addition to information that can be retrieved from a package's control file, embodiments of the invention provide the user with further detailed information about a software package by querying its manifest file. A manifest file typically contains detailed information ("meta-information") about files that are condensed and packaged together to form a software package such as package 200.
>
> In one or more embodiments of the invention, package 200 is automatically associated with a manifest file upon creation. A manifest file, for example, can contain a list of all files contained in payload file 210 that make up package 200, their names, the directory they are stored in, dependencies there between, and other pertinent information necessary to access and manipulate these files.
>
> The manifest can be dynamically generated from the compressed cpio archive of an uninstalled package, for example. In embodiments of the invention, the dynamically generated manifest can be stored in a local archive file at the same time that a package is installed on a computer system. This way package information can be queried even prior to the installation of the package.

These cited portions merely disclose contents of a software package that may be used by a target system. None of these cited portions refer to an element that can be equated to the recited master node. Therefore, *Foster* cannot even suggest that a master node performs the actions of

comparing and determining. In contrast, according to Claim 3, the master node **compares** desired characteristics to each module in a software update **to determine** which modules should be sent to a node.

Due to the fundamental differences already identified and to expedite the positive resolution of this case, a separate discussion of the other additional limitations is not included at this time. Representatives for the applicants reserve the right to further point out the differences between the cited art and the novel features recited in the dependent claims.

## II. CONCLUSIONS & MISCELLANEOUS

For the reasons set forth above, all of the pending claims are now in condition for allowance. The Examiner is respectfully requested to contact the undersigned by telephone relating to any issue that would advance examination of the present application.

A petition for extension of time, to the extent necessary to make this reply timely filed, is hereby made. If applicable, a law firm check for the petition for extension of time fee is enclosed herewith. If any applicable fee is missing or insufficient, throughout the pendency of this application, the Commissioner is hereby authorized to any applicable fees and to credit any overpayments to our Deposit Account No. 50-1302.

Respectfully submitted,
HICKMAN PALERMO TRUONG & BECKER LLP

Dated: May 2, 2008

/DanielDLedesma#57181/
Daniel D. Ledesma
Reg. No. 57,181

2055 Gateway Place Suite 550
San Jose, California 95110-1083
Telephone No.: (408) 414-1080 ext. 229
Facsimile No.: (408) 414-1076